Preprint version. Published in Computers and Fluids **125**, 11-24 (2016).
DOI: 10.1016/j.compfluid.2015.10.019

On the efficiency of gradient based optimization algorithms for DNS-based optimal control in a turbulent channel flow

C. Nita[a], S. Vandewalle[b], J. Meyers[a,*]

[a]*Department of Mechanical Engineering, KU Leuven, Celestijnenlaan 300A, B3001 Leuven, Belgium*
[b]*Department of Computer Science, KU Leuven, Celestijnenlaan 200A, B3001 Leuven, Belgium*

Abstract

We analyse the performance of different limited-memory quasi-Newton methods for unconstrained DNS-based optimization. Optimization based on Direct Numerical Simulation (DNS) of turbulent flows is extremely expensive, as functional and gradient evaluations require the simulation of Navier–Stokes and adjoint Navier–Stokes equations with high space and time resolution. Nowadays, simple and robust nonlinear conjugate gradient methods are generally used for DNS-based optimal control, as they do not require much memory overhead in a large control space. In the current study, we investigate the use of quasi-Newton methods instead. They combine a cheap approximation of the Hessian to improve step direction and step length, leading to faster convergence of the optimization. Since control spaces are often large in DNS-based optimization, we investigate only limited-memory quasi-Newton methods. Three methods are studied, i.e., the discrete truncated Newton method, the limited-memory BFGS method, and the damped L-BFGS method. The latter method is designed for constrained optimization, but can also address unconstrained problems. Furthermore, the damped L-BFGS method only requires the Armijo condition in the line search, not the Wolfe conditions, limiting expensive functional and gradient evaluations. We investigate the combination of the three quasi-Newton methods with three different line-search methods either based on bisection, quadratic interpolation, or cubic interpolation. Initially, all possible combinations are evaluated in a test problem that is based on the extended

*Corresponding author
Email addresses: cornelia.nita@kuleuven.be (C. Nita), stefan.vandewalle@cs.kuleuven.be (S. Vandewalle), johan.meyers@kuleuven.be (J. Meyers)

Rosenbrock function. The three best performing methods are further tested in two different DNS-based optimal control cases in a turbulent channel flow at $Re_\tau = 180$. This reveals that the damped L-BFGS method in combination with a cubic line search performs best, closely followed by classical L-BFGS with cubic line search. Though the damped L-BFGS often requires a few more iterations to reach convergence, this is compensated by a more cost effective line search, with fewer functional and gradient evaluations. Moreover, compared to the conjugate-gradient method, damped L-BFGS speeds up convergence by a factor of four.
Keywords: DNS-based optimization, quasi-Newton method, limited-memory BFGS, damped L-BFGS, truncated Newton method, turbulent flow

1. Introduction

In recent years, the use of adjoint-based optimal control of Direct Numerical Simulations (DNS) or Large-Eddy Simulations (LES) has gained increased interest in areas such as drag reduction in turbulent boundary layers [1], noise reduction [2–4], turbulent mixing [5–7], or wind-farm energy extraction [8]. The main challenge for these types of optimal control studies is the large computational costs involved for the optimization of the controls. This requires multiple function evaluations that are based on a full DNS or LES, in combination with adjoint-based gradient evaluations, making optimal control several orders of magnitude more expensive than a standard DNS or LES. Nowadays, the method of choice in these applications is the Polak–Ribière nonlinear conjugate-gradient method in combination with a Brent line search, as first used by Bewley *et al.* [1] in the context of DNS-based optimal control. In the current study, we focus on a comparison of a number of alternative gradient-based optimization methods, concentrating on the total cost in terms of the required number of direct numerical and adjoint simulations for convergence of the optimization problem. To that end, we concentrate on a range of quasi-Newton methods, including the limited-memory BFGS method [9], the damped limited-memory BFGS method [5], and the truncated Newton method [10], all combined with various line-search algorithms.

DNS-based optimal control involves optimization problems that are constrained by the Navier–Stokes equations (see, e.g., [11–14]). In some PDE-constrained optimization prob-

lems, the partial differential equations are included explicitly as a constraint in the optimization formulation (see [15] for a discussion). However, in case of DNS (or LES), the state space is too big for this to be practicable, and the optimization cost functional is formulated in a reduced form in which the direct numerical simulation is used as an implicit function to evaluate the effect of the controls on the objective functional [16, 17]. The gradients of the reduced cost functional are evaluated using an adjoint DNS. In this setting, DNS-based optimization problems are in most cases unconstrained, i.e. in view of cost and complexity, no additional constraints on controls or states are presented (exceptions are, e.g., found in Refs. [5–7]). Therefore, in the current work, we investigate the efficiency of unconstrained DNS-based optimization problems only.

The efficiency of optimization algorithms is measured by their ability to find the optimal solution in a reasonable amount of time and can be assessed by evaluating the computational cost per iteration and the number of iterations required. In DNS-based optimization, the dominant cost is related to the repeated simulations of the Navier–Stokes equations, and adjoint Navier–Stokes equations for evaluations of the cost functional and its gradient. Therefore, efficiency is directly related to the number of functional and gradient estimations necessary in the optimization process. Moreover, given the large computational cost of DNS-based optimal control, the optimization algorithm is often stopped well before convergence, so that the number of functional and gradient evaluations per unit of cost functional improvement are also important. In this context, non-linear conjugate-gradient and steepest-descent methods are well suited, as they are simple in implementation, and efficient when only a few iterations are considered. However, these methods yield information on the step direction, but do not provide a step size estimate. To that end, a line search is used. For the Polak–Ribière non-linear conjugate-gradient method, a new descent direction is only guaranteed when the line search is converged to its minimum [18], often requiring many iterations (note that, e.g., an adaptation of the Wolfe conditions [18] or different non-linear conjugate gradient versions [19, 20] alleviate this problem). It is well known that the use of curvature information, next to gradient information, in quasi Newton methods can significantly speed

up optimization algorithms [18]. In contrast to non-linear conjugate-gradient methods, such methods do provide a step-length estimate.

The use of Newton methods in gradient-based optimization requires the Hessian of the cost function. In the context of DNS-based optimal control, the direct evaluation of the Hessian is computationally infeasible. Moreover, in control spaces with a large number of degrees of freedom, the storage in memory can become prohibitive. Instead, limited-memory representations of the true Hessian can be approximated using a recursion formula, such as the so-called limited-memory BFGS method [18]. In addition, Hessian-free inexact Newton methods are often considered in this context [21, 22]. In nonlinear problems, the step length provided by (quasi-)Newton methods does not always guarantee a descent direction in the next iteration, so that also for these algorithms, an additional line search is sometimes required. The efficiency of this line search method is also of importance. In the limited-memory BFGS (L-BFGS) method, the step length has to satisfy the strong Wolfe conditions to ensure convergence of the algorithm [18]. To check these conditions, an additional function and gradient evaluation are required for every iteration in the line-search method. Therefore, we will also consider a damped L-BFGS method [5] in the current work. Although damped L-BFGS methods are designed for constrained optimization problems (e.g. in combination with an SQP algorithm), they can also be used in unconstrained problems, having the advantage that the step length in the line search only needs to satisfy the Armijo condition, thus requiring only one additional function evaluation for every line-search iteration. With respect to line search algorithms, we investigate bisection methods, quadratic line search methods, and cubic-quadratic methods following the implementation guidelines from Nocedal [18], Powell [23], Moré–Thuente [24], and Dennis and Schnabel [25].

Optimization algorithms are often benchmarked against a range of standard test functions. Classical examples are, e.g., found in Nash and Nocedal [10], or Zou *et al.* [26]. For instance, Nash and Nocedal [10] established for a set of 45 different test problems that the L-BFGS method was more efficient in terms of function evaluations for highly nonlinear problems compared to the Polak–Ribière conjugate gradient method, and a truncated New-

ton method. However, testing of such algorithms in DNS-based optimal control has never been performed. In the current work, we follow a two-stage approach. First, we test the different algorithms and their combination with line search methods based on a standard optimization benchmark, i.e. the extended Rosenbrock function. From this, we select the best combination of the different quasi-Newton methods with the line search methods, and test these for two different types of optimal control problem in a turbulent channel flow.

The work is organized as follows. Section 2 provides the details of the optimization problem for the turbulent channel flow. Section 3 describes the algorithms used in this study. The results for the two test optimization cases (Rosenbrock function and turbulent channel flow) are presented in section 4. Finally in section 5, we present some conclusions.

2. DNS-based optimal control of a turbulent channel flow

In the current section, we focus on the formulation of two different DNS-based optimization problems that are relevant for turbulent boundary layers, and will be used for the evaluation of the different optimization algorithms in § 4.2. To that end, we select a simple wall-bounded flow, i.e. turbulent channel flow. The optimal control theory for turbulent wall-bounded flows by means of DNS (or LES) has been investigated among others with optimization cost functionals aiming at a decrease of turbulent kinetic energy and drag [1], a reduction of the skin friction [27], or an increase of total extracted energy by internal forces [8].

The first optimal control problem that we consider aims at maximizing energy extraction by optimizing a distributed volume force in the channel. Such an optimal control problem is, e.g., relevant in optimization of wind-farm boundary layers, where the turbine power extraction and thrust force can be dynamically steered in space and time [8]. In the current work, we omit the details of turbine modeling, and compose the problem using simpler volume forces in a subregion of the domain. A second optimal control problem that is considered, aims at reducing turbulent kinetic energy, typically with the indirect objective to reduce total drag [1]. Here we select a control mechanism that is similar to the work by Mamory and Fukagata [28]. Remark that both optimal control problems are somewhat opposite in

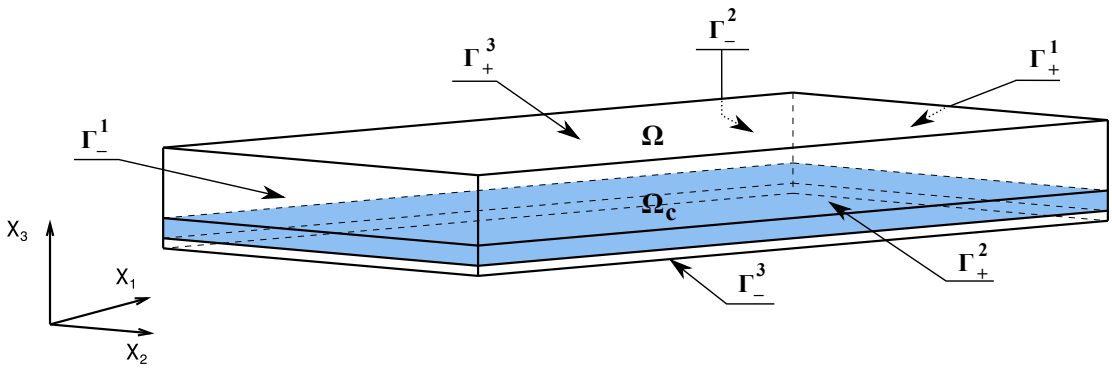

Figure 1: Computational domain

terms of turbulent flow physics. Where drag reduction is here achieved by reducing turbulent kinetic energy as much as possible, increasing energy extraction requires sufficient turbulence levels, as these are important for the transport of kinetic energy towards the forcing region [8, 29].

Thus, for the first case, we consider the optimization of power extraction from a fully developed turbulent channel flow by volume force f distributed over a volume $\Omega_c \subset \Omega$, with Ω the full channel-flow domain (cf. Figure 1 for details). The cost functional is then given by:

$$\mathcal{J} = -\int_0^T \int_{\Omega_c} f \cdot u \ \mathrm{d}x\mathrm{d}t + \mu \mathcal{T}(f,u), \tag{1}$$

where the first part corresponds to the power extracted from the boundary layer over an optimal control horizon T, and $\mathcal{T}(f,u)$ is a penalization term (cf. further discussion below), weighted with a factor μ. The control force $f(x_3,t)$ is a stream-wise body force, which in the current study does not depend on x_1 or x_2, and equals to zero in $\Omega \setminus \Omega_c$.

For the second case, we consider the reduction of turbulent kinetic energy in the turbulent channel flow. The problem is formulated in terms of the terminal kinetic energy at the end of the optimal control time horizon (Similar to Refs. [1, 30]). Thus,

$$\mathcal{J} = \int_\Omega \left\| u(x,T) - \langle u(x,T) \rangle \right\|^2 \mathrm{d}x, \tag{2}$$

where $\langle u(x,T) \rangle$ is the horizontally averaged velocity field. In this case, the control force f

is a wave-like wall-normal body force, formulated as [28]

$$f = \exp(-\frac{x_3^+}{\Delta^+}) f(kx_1, t), \tag{3}$$

with a fixed penetration length $\Delta^+ = 12.5$ in wall units (following the parametric study of Mamori and Fukagata [28]). Note that the forcing is only added at the bottom wall.

2.1. Problem and reduced problem formulation

Given the cost functionals above, we arrive at the following optimization problem:

$$\min_{f,u} \mathcal{J}(f, u), \tag{4}$$

subject to

$$\frac{\partial u}{\partial t} + (u \cdot \nabla)u + \nabla p - \frac{1}{Re}\nabla^2 u = \tilde{f} e_f \quad \text{in } \Omega \times (0, T], \tag{5}$$

$$\nabla \cdot u = 0 \quad \text{in } \Omega \times (0, T], \tag{6}$$

$$u|_{t=0} = u_0 \quad \text{on } \Omega, \tag{7}$$

$$u = 0 \quad \text{on } \Gamma_3^+ \cup \Gamma_3^-, \tag{8}$$

where e_f is the unit vector e_1 in the first optimal control case, and e_3 in the second case. The primal (forward) problem is governed by incompressible Navier-Stokes equations (Eq. 5, 6) and a proper set of boundary conditions on the walls (Eq. 8), where u is the velocity vector and p is the pressure. Periodic boundary conditions are imposed in the streamwise (x_1) and the spanwise(x_2) directions and no-slip in the normal direction (x_3).

To solve the optimization problem, the problem is considered in its reduced form (cf. also discussion in introduction). The Navier-Stokes equations, denoted symbolically by $\mathcal{P}(f, u) = 0$, are not formulated as an explicit constraint to the optimal control problem above (Eq. 4–8), but they are satisfied at every iteration of the optimization process [16]. This leads to

$$\min_f \hat{\mathcal{J}}(f) \tag{9}$$

with $\hat{\mathcal{J}}(f) = \mathcal{J}(f, u(f))$ and where $u(f)$ indicates the solution of the primal equations at every perturbation of the control f, such that $\mathcal{P}(f, u(f)) \equiv 0$.

2.2. Continuous adjoint technique

The iterative process of the optimization using gradient information requires the evaluation of the cost functional Jacobian with respect to small changes in the design variables $(f + \delta f)$:

$$\hat{\mathcal{J}}'(f, \delta f) = \left(\frac{\partial \mathcal{J}(f,u)}{\partial u}, \delta u\right) + \left(\frac{\partial \mathcal{J}(f,u)}{\partial f}, \delta f\right), \tag{10}$$

where $(\cdot, \cdot)$, the L^2 inner product employed for the representation of the derivative, is defined as:

$$(u, v) = \int_0^T \int_\Omega u \cdot v \ \mathrm{d}\Omega \mathrm{d}t. \tag{11}$$

The variable $\delta u(f+\delta f)$ represents the sensitivity of the turbulent flow solution to all possible arbitrary changes (δf) in the distributed control parameters (f). In the context of the optimization coupled with computational fluid dynamics (CFD) the approximation of the directional derivative by finite difference is prohibitive. Consequently, a tractable procedure for the flow control problems is to calculate the sensitivity of the performance functional $(\hat{\mathcal{J}})$ using the adjoint method with a computational cost no greater than solving one forward simulation (DNS)(see e.g. [11, 13, 31]).

In order to derive the continuous adjoint equations, and the cost functional gradient in terms of the adjoint solution, the formal Lagrangian approach can be employed, in which the gradient of the reduced problem is derived from the variation of the Lagrangian of the original problem (Eq. 4–8) with respect to control variables. The adjoint equations follow from the variation of the Lagrangian to the state. The procedure is well known, and can be, e.g., found in Refs. [8, 16, 32]. Applied to the Navier–Stokes equations this leads to following adjoint equations (cf., e.g., Ref. [31])

$$-\frac{\partial u^*}{\partial t} + (\nabla u)^T u^* - (u \cdot \nabla) u^* + \nabla p^* - \frac{1}{Re}\nabla^2 u^* = f^* \tag{12}$$

$$-\nabla \cdot u^* = 0 \tag{13}$$

with u^*, p^* and f^* the adjoint velocity, pressure and source term respectively.

For the first optimization problem, we further find

$$f^* = \frac{\partial \mathcal{J}}{\partial u} = -f + \mu \frac{\partial \mathcal{T}}{\partial u} \tag{14}$$

The adjoint initial condition, referred to as 'terminal' condition as it is defined at the time horizon T, corresponds to

$$u^*(x_3, T, f) = \frac{\partial \mathcal{J}}{\partial u_T} = \frac{\partial \mathcal{T}}{\partial u_T}, \tag{15}$$

where $u_T = u(x, T)$. The gradient of the reduced cost functional is formally defined as the Riesz representation of the directional differential [16], i.e.

$$\hat{\mathcal{J}}'(f, \delta f) \equiv (\nabla \hat{\mathcal{J}}, \delta f) \tag{16}$$

Finally, the gradient of the reduced cost functional can be expressed in term of the adjoint variable estimated at the current value of the control [16, 17], leading to:

$$\nabla \hat{\mathcal{J}} = \frac{\partial \mathcal{J}}{\partial f} + \left[\frac{\partial \mathcal{P}}{\partial f}\right]^* u^* = -u + \frac{\partial \mathcal{T}}{\partial f} + u^* \tag{17}$$

Similarly, for the second problem, we find

$$f^* = 0, \tag{18}$$

$$u^*(x, T, f) = \frac{1}{\Omega}\left(u(x, T, f) - \langle u(x, T, f)\rangle\right), \tag{19}$$

$$\nabla \hat{\mathcal{J}} = u^*. \tag{20}$$

Finally, in this manuscript, we use an continuous adjoint methodology, that takes above continuous adjoint equations and subsequently discretizes them (see next section). It is well known that an alternative to this approach is to first discretize the forward system, and then formulate the discrete adjoint equations from the discrete forward system (cf. e.g., [33]). In this case, sensitivities of the cost functional evaluated with the (linearized) forward system equal sensitivities evaluated with the adjoint system up to machine accuracy. In a continuous adjoint approach, this is not guaranteed. In recent years, a number of studies have shown the possible advantage of discrete adjoint methods for fluid mechanics applications [34–37]. In the current work, we do not want to advocate one method over the other, in particular since also continuous approaches have been very useful in the past [1, 4, 8]. However, when using a continuous approach, it is important to verify that the inconsistency between forward and backward sensitivity is small. In §2.4, we show that the relative error between our adjoint-based sensitivity and the forward sensitivity is in the order of 10^{-8}.

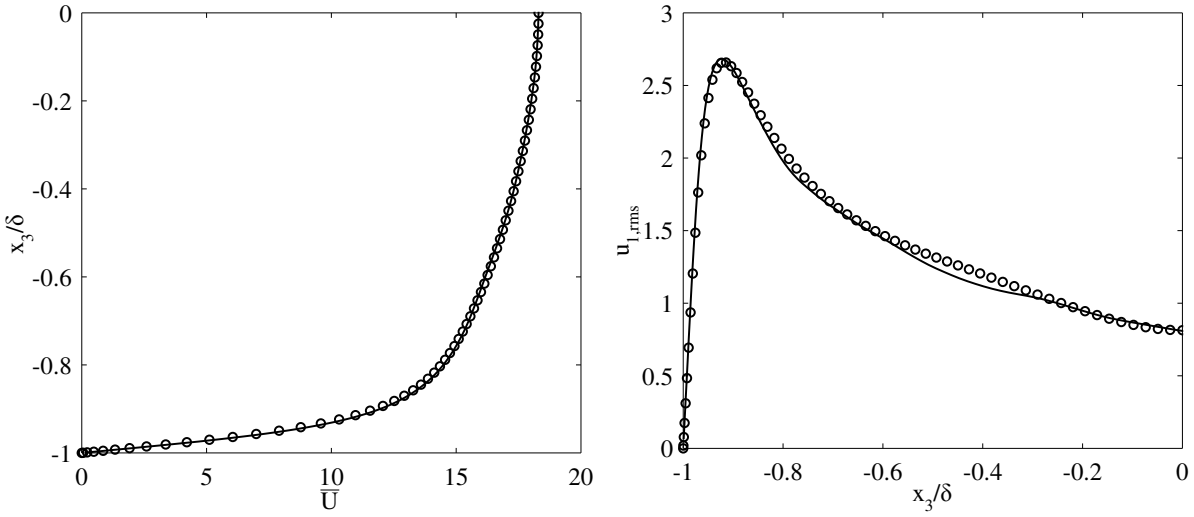

Figure 2: Verification of the mean velocity profile ($\overline{U}$) (—) (left) and the root mean velocity fluctuations $u_{1,rms}$ (—) (right) with the database results (∘) from [40, 41].

2.3. Numerical method

The forward problem, i.e. DNS of incompressible turbulent channel flow, is performed in our tests with the mixed pseudo-spectral and finite difference code SP-Wind (cf. e.g., [5–7]). A fourth-order energy-conserving finite difference scheme [38] is used in the wall normal direction, and a pseudo-spectral spectral method with Fourier basis functions [39] is implemented in the streamwise and spanwise directions, and dealiasing is performed using the 3/2 rule. Time integration is based on a standard fourth-order explicit Runge–Kutta scheme.

The adjoint equations are integrated backward in time and discretized using the same approach and the same grid as for the forward equations. The state variable (velocity) is stored at every time step and at every grid point during the forward simulation, and is used in the adjoint simulations. Details of the implementation are found in [6].

For the first optimal control case, we select the size of the computational box Ω as $L_{x_1}\times L_{x_2}\times L_{x_3} = 4\pi\delta\times 2\pi\delta\times 2\delta$ with a grid size of $N = 128\times 128\times 200$. The Reynolds number $Re_\tau = 180$ is based on the friction velocity $u_\tau \equiv (\delta\partial p_\infty/\partial x)^{1/2}$ and the channel half-width

$\delta(=1)$, and where $\partial p_\infty/\partial x$ is the driving pressure gradient in the channel. The grid in the wall-normal direction x_3 is stretched using a tangent hyperbolic function. The time steps δt are restricted by setting the convective and diffusive Courant-Friedrichs–Lewy (CFL) number to 0.4. The computational mesh and time steps are sufficiently fine for grid-converged DNS, and this is verified by comparing DNS without source term to the DNS data of Kim et al. [40, 41] (cf. Figure 2). The simulations are initialized using a quadratic streamwise velocity profile to which random perturbations are added. Subsequently, simulations are progressed over 69 through-flow times to spin up realistic turbulence. The turbulent velocity field at the end of this spin up is used as the real initial conditions for simulations and optimal control. Finally, the control forces are located in the volume $\Omega_c = L_{x_1} \times L_{x_2} \times [-0.8\delta, -0.4\delta]$ of length l_{x_3} in the normal direction.

For the second optimization case, the size of the computational domain is $4\pi\delta \times \pi\delta \times 2\delta$ with a finer grid in the streamwise and spanwise direction $192 \times 128 \times 160$. The numerical set-up is similar to the previous case and the flow is driven by a constant pressure gradient in order to maintain a proper wall-unit scale [42] with $Re_\tau = 180$. Furthermore, in the x_1 direction, we represent the control on the first 48 Fourier modes only in our pseudo-spectral method (instead of the 96 modes that would be allowed by the discretization).

2.4. Adjoint gradient validation

Finally, in order to quantify the error between the adjoint gradient and the forward gradient in our method, we compare the adjoint sensitivity to a forward sensitivity that is obtained using a finite-difference discretization of the first part of Eq. (10), i.e.

$$(\nabla \hat{\mathcal{J}}_1, \delta f(x_3, t)) = \frac{\mathcal{J}(f_0, u(f_0 + \alpha\delta f)) - \mathcal{J}(f_0, u(f_0))}{\alpha} + \mathcal{O}(\alpha^2), \tag{21}$$

where we will make a comparison for different step lengths α. Since the forward evaluations require one function evaluation per control change, a limited number of changes δf is considered, for which we select

$$\delta f(x_3, t) = \frac{A}{4}\left(1 + \cos\left[\frac{n\pi}{\ell_{x_3}}(x_3 + 0.4)\right]\right)\left(1 + \sin\left[\frac{n\pi}{T}t\right]\right), \tag{22}$$

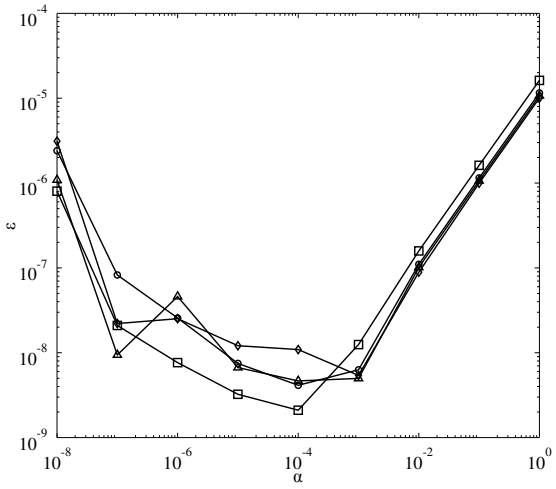

Figure 3: The relative error $\epsilon = |\delta\hat{\mathcal{J}}_{1,FD} - \delta\hat{\mathcal{J}}_{1,AD}|/|\hat{\mathcal{J}}_{1,FD}|$ at different step lengths α for $n = 1$ ($\square$), $n = 2$ ($\triangle$), $n = 3$ ($\circ$) and $n = 4$ ($\Diamond$).

where A is a normalization such that $\|\delta f\| = 1$, and $n = 1$, 2, 3, or 4. Further, we take $f_0(x_3, t) = 0.2$.

Figure 3 shows the relative error between the forward and the adjoint gradient for different step lengths. It is appreciated that for $\alpha = 10^{-4}$ a minimum relative error in the order of 10^{-8} is observed. For lower values of α the error increases. This is related to the decreasing arithmetic accuracy of the finite-difference approximation in Eq. (21). In fact, we find for all four perturbations δf, that $\delta\hat{\mathcal{J}}_1/|\hat{\mathcal{J}}_1| \sim \alpha/100$, so that the arithmetic precision of the difference in Eq. (21) is in the order of 10^{-9} at $\alpha = 10^{-4}$, and 10^{-7} at $\alpha = 10^{-6}$ (given a machine accuracy of 10^{-15}). For values of alpha larger than 10^{-3} we observe that the relative error also increases significantly. This is related to nonlinear effects that start to play a role in the forward evaluation, since to that end, we solve the full non-linear Navier–Stokes equations (cf. Eq. 21), not the forward linearized equations as given in the first part of Eq. (10). We conclude that the relative error between forward and backward sensitivities is in the order of 10^{-8}. In theory, this error might be even smaller (since it still contains the finite difference error in the approximation of Eq. 10), but this can only be verified by also solving the forward linearized sensitivity problem, which we do not have available in

our code. Moreover, in practice, it is not important to have an adjoint gradient accuracy that is lower than that of Eq. (21), since the latter is the effective accuracy with which cost functional values are compared during the line search in the optimization algorithm.

3. Unconstrained optimization methods

In this section we briefly review the optimization algorithms that are tested in the current study: the Polak–Ribière conjugate gradient algorithm, the limited memory BFGS method, the damped L-BFGS method and the discrete truncated Newton algorithm in combination with the L-BFGS Hessian approximation.

All methods are iterative methods that update the controls from iteration k to $k+1$ using $f_{k+1} = f_k + \alpha d_k$, where the vector d_k provides the step direction, which is scaled with α to yield a good step length. Newton-based methods, estimate the correct size of d_k, so that α is often equal to one; otherwise, a line-search method is used to iteratively determine the size of α. Conjugate-gradient methods do not properly scale d_k, so that a line search is almost always required.

3.1. Nonlinear conjugate gradient method

The nonlinear conjugate gradient method is one of the first techniques that was used in computational optimization governed by partial differential equations. It is widely used in solving large-scale optimization problems having as major advantage limited storage requirements [18]. The Polak–Ribière variant is to-date still the standard for DNS and LES-based optimization. Here we briefly review the method, as it will be used as a reference, next to the different quasi-Newton methods, in §4.2 for the DNS-based optimal control of a turbulent channel flow.

The Polak–Ribière conjugate gradient method determines the step direction as [18, 43]

$$d_k = \begin{cases} -\nabla \hat{\mathcal{J}}_k & k = 0, \\ -\nabla \hat{\mathcal{J}}_k + \beta_k d_{k-1}, & k \geq 1, \end{cases} \tag{23}$$

where the choice for the scalar β_k is given by

$$\beta_k = \frac{\nabla\hat{\mathcal{J}}_k^T\left(\nabla\hat{\mathcal{J}}_k - \nabla\hat{\mathcal{J}}_{k-1}\right)}{||\nabla\hat{\mathcal{J}}_{k-1}||^2} \tag{24}$$

In order to guarantee that d_{k+1} remains a descent direction, the method requires a line-search to find the α that minimizes the cost function in the direction d_k. To that end, derivative-free line-search methods are usually preferred. In the context of DNS-based optimization with the Polak–Ribière method, the Brent line-search algorithm has been often used [1, 7]. Details of the algorithm are found in Ref. [44].

3.2. Limited memory BFGS method

For large-scale problems, when the Hessian is expensive to evaluate, the L-BFGS algorithm is a popular quasi-Newton method that is often used. Because the approximation to the inverse Hessian H_k of the objective function is often too large to store, only m vector pairs, s_k and y_k that define the matrix implicitly, are saved.

A method to initialize the algorithm, at every iteration k, is to first define the diagonal matrix [18] $H_k^0 = \gamma_k I$ with

$$\gamma_k = \frac{y_{k-1}^T s_{k-1}}{||y_{k-1}||_2^2}, \tag{25}$$

and

$$s_k = f_{k+1} - f_k, \qquad y_k = \nabla\hat{\mathcal{J}}_{k+1} - \nabla\hat{\mathcal{J}}_k. \tag{26}$$

At $k = 1$, $\gamma = 1$ is used. The approximation H_k to the inverse Hessian is then constructed by applying m corrections to H_k^0, using the relation:

$$H_k = W_m^T H_k^0 W_m + \sum_{i=2}^{m} \frac{1}{y_{k-i}^T s_{k-i}} W_{i-1}^T s_{k-i} s_{k-i}^T W_{i-1} + \frac{1}{y_{k-1}^T s_{k-1}} s_{k-1} s_{k-1}^T, \tag{27}$$

with

$$W_i = \prod_{j=1}^{i} \left[I - 1/\left(y_{k-j}^T s_{k-j}\right) y_{k-j} s_{k-j}^T\right]. \tag{28}$$

The matrix H_k is never explicitly determined or stored, but using above formulation, the step direction

$$d_k = -H_k \nabla \hat{\mathcal{J}}_k \tag{29}$$

can be determined using a two-loop recursion algorithm using the m vector pairs, s_k and y_k [18].

In order to guarantee a positive definite H_{k+1}, and descent direction d_{k+1}, the step αd_k should satisfy the strong Wolfe conditions

$$\hat{\mathcal{J}}(f_k + \alpha d_k) \leq \hat{\mathcal{J}}(f_k) + c_1 \alpha \nabla \hat{\mathcal{J}}_k^T d_k, \tag{30}$$

$$\left| \nabla \hat{\mathcal{J}}(f_k + \alpha d_k)^T d_k \right| \leq -c_2 \nabla \hat{\mathcal{J}}_k^T d_k, \tag{31}$$

where we use the constants $c_1 = 10^{-4}$ and $c_2 = 0.9$ as suggested in [18]. Since Newton methods properly scale d_k (for quadratic problems), $\alpha = 1$ often directly satisfies these conditions. Otherwise, a line-search method is required that finds α satisfying these conditions. As verification of the second condition requires the evaluation of the gradient at $f_k + \alpha d_k$, this leads to one additional function evaluation and gradient evaluation per line-search step.

We investigate three different line-search techniques, i.e., two variants of the Nocedal method [18] described on pag. 60–61 (precise implementation details are given in Appendix B), and the Moré–Thuente method [24]. In a first phase, they all bracket an interval containing suitable points; in a second step they respectively use a bisection method, a bisection-quadratic line search, and a cubic-quadratic line search to shorten this interval.

3.3. Damped L-BFGS method

The damped L-BFGS method, is a limited-memory BFGS method that can be used for constrained optimization problems. For such problems, the standard (limited-memory) BFGS method does not guarantee a positive definite matrix H_k. The damped L-BFGS method uses 'Powell's trick' [23] to remedy this fact and replaces s_k in (Eq. 27, 28) by [5]

$$r_k = \theta_k s_k + (1 - \theta_k) H_k y_k, \tag{32}$$

with

$$\theta_k = \begin{cases} 1 & s_k^T y_k \geq 0.2 y_k^T H_k y_k \\ \frac{0.8 y_k^T H_k y_k}{y_k^T H_k y_k - s_k^T y_k}, & s_k^T y_k < 0.2 y_k^T H_k y_k \end{cases}, \tag{33}$$

and $s_k = f_{k+1} - f_k$. In contrast to the classical L-BFGS method, the step length αd_k obtained from damped L-BFGS only needs to satisfy the first Wolfe condition (Eq. 30), also called the Armijo condition for guaranteeing H_{k+1} positive definite (and d_{k+1} a descent direction). Thus, whenever $\alpha = 1$ does not directly satisfy the Armijo condition, the additional cost per line-search iteration amounts to one function evaluation, but no extra gradient evaluations. Moreover, since only one condition needs to be satisfied, the line search may also converge faster.

The damped L-BFGS method can be straightforwardly used for unconstrained optimization problems also. Even though the use of r_k may lead to a less accurate representation of the inverse Hessian, this may be offset in highly non-linear problems by gains in the line search algorithm. Again, we will test three line search algorithms. After a first bracketing stage, they either use a bisection method (a simple bisection-based backtracking line search [18, 45]), a quadratic line search (the Powell method [23]), or a cubic-quadratic line search (Dennis and Schnabel [25], see algorithm described on page 126).

3.4. Discrete truncated Newton method

In contrast to BFGS methods, where approximations to the Hessian are constructed based on gradients obtained over a number of iterations, the truncated Newton methods aims at using the local Hessian to find the search direction. A standard Newton method finds the search direction from

$$\nabla^2 f(x_k) p_k = -\nabla f(x_k). \tag{34}$$

In a truncated Newton method, this system is solved iteratively using a preconditioned linear conjugate gradient method, that is truncated after a few iterations only [21, 22] (note that a linear conjugate-gradient method is a standard iterative tool for solving large linear

systems, and should not be confused with a nonlinear conjugate-gradient method such as the Polak–Ribiére method). In this way, second derivatives are only required in the subsequent conjugate gradient directions, and can be evaluated using finite-differences. This requires one function evaluation and one gradient evaluation per conjugate gradient iteration. Numerous investigations of different variants of the inexact Newton method have been performed in the context of large scale optimization problem [10, 26]. The advantage of the truncated Newton method – also called the Hessian-free inexact Newton method – is that it converges to a minimum with a smaller number of iterations compared with the L-BFGS method. However, the drawback of this method is the significant cost in terms of functional and gradient evaluations required per iteration.

In practice, the truncated Newton method is often combined with an L-BFGS method [46, 47]. For strongly nonlinear regions, where the truncated-Newton search direction is not a descent direction or when the step-size is poorly predicted, the L-BFGS step is used instead. In the current work, we test such a hybrid approach. The implementation follows Ref. [47] and [18] (page 169). In the current study, we use a maximum of 2 conjugate gradient steps, preconditioned by a limited memory BFGS algorithm, before truncating the Newton method. Further details of the algorithm are provided in Appendix A.

4. Results

First of all, we evaluate the combination of the three quasi-Newton methods discussed above with three different line-search methods using a test case based on the extended Rosenbrock function. Results are presented in §4.1. The best combinations are selected for further testing in DNS-based optimal control of a turbulent channel flow. This is further discussed in §4.2.

4.1. Rosenbrock function

The extended Rosenbrock function [48] is a classical and challenging problem for unconstrained optimization, that allows for a fast evaluation of optimization algorithms mimicking

Table 1: Combination of quasi-Newton methods and line-search methods

	Bisection	Quadratic	Cubic-quadratic
damped L-BFGS	[18, 45]	Powell [23]	Dennis and Schnabel [25]
L-BFGS	Nocedal-bisection [18]	Nocedal-quadratic [18]	Moré–Thuente [24]
dTN+L-BFGS	Nocedal-bisection [18]	Nocedal-quadratic [18]	Moré–Thuente [24]

problems with large parameter spaces. The function is given by

$$f(x) = \sum_{i=1}^{N/2} \left[100 \left(x_{2i} - x_{2i-1}^2 \right)^2 + \left(1 - x_{2i-1} \right)^2 \right], \tag{35}$$

where $x \in \mathbb{R}^N$. There is one global minimum at $x^* = (1, \ldots, 1)$, with $f(x^*) = 0$.

Here we use this problem to select the best combinations of quasi-Newton methods and line-search methods. Three different line searches are tested, i.e. a bisection, a quadratic, and a cubic-quadratic method. An overview of the possible combinations is provided in Table 1. It is appreciated that the line search methods differ between damped L-BFGS and L-BFGS, but this is related to the fact that these methods need to satisfy different conditions (i.e. the Armijo and Wolfe conditions respectively). Finally note that all line search algorithms start with a bracketing phase. Further details are found in the respective references.

In order to mimic large optimization problems, three different dimensions $N_1 = 10^3$, $N_2 = 10^4$ and $N_3 = 10^5$ are selected for testing. The optimization problem is stopped for all cases when

$$||\nabla f(x)|| \leq 10^{-4} \tag{36}$$

is satisfied. Various different starting points are selected as further discussed below.

First of all, in Table 2, the number of functional and gradient evaluations ($F+G$), and the number of iterations (it) are reported for three different deterministic starting points, with a different distance from the optimum. The first point x_0 is constructed with all elements with even index x_i (i even) equal to 1, and all elements with odd index equal to -1.2. For

sake of compactness, we use the short-hand notation $x_0 = [x_{i=odd}, x_{i=even}] = [-1.2, 1]$. Two more points correspond to $10 \times x_0$; and $100 \times x_0$. Note that the point x_0 is a standard starting point for testing with the extended Rosenbrock function [25, 49]. Later, we will further focuss on random starting points (but all with roughly the same norm).

It is appreciated from Table 2 when looking at function and gradient evaluations, that the discrete Truncated Newton method is always outperformed by the L-BFGS and damped L-BFGS methods. The dTN method does compete in terms of number of iterations, but the additional overhead of function and gradient evaluations per iteration, makes the method significantly less efficient. When looking at the L-BFGS method, the combination with the quadratic line search is the least efficient, while for the damped L-BFGS method, the bisection line-search is least efficient. For the BFGS methods, the number of function and gradient evaluations does not strongly depend on the degrees of freedom N, but there is a weak dependence observed on the norm of the starting point, e.g. for the damped L-BFGS method in combination the with Dennis and Schnabel line search, $F + G$ increases from roughly 60 to 150 to 300 for x_0, $10x_0$, and $100x_0$.

In Figure 4(left) the cumulative sum of function and gradient evaluations is plotted against the number of iterations for the 9 different quasi-Newton methods using x_0 and $N = 10^3$. In Figure 4(right) the corresponding decrease of the Rosenbrock function is shown. The figure shows that the differences observed in Table 2 accumulate gradually during iterations. The case reported here illustrates that the L-BFGS method requires expensive iterations in terms of functional and gradient evaluations to reach the optimum values while the damped version needs more iterations but less expensive ones. Furthermore, the hybrid discrete Truncated Newton with L-BFGS algorithm, is clearly outperformed by the quasi-Newton methods.

In order to also evaluate the quasi-Newton methods for starting points that have a different orientation than x_0, fifty random sequences are generated with numbers in the interval $[-2, 2]$. This is again done for three different dimensions, i.e. $N = 10^3$, 10^4, and 10^5. These starting points are then used to test the different algorithms, and the average number of

Table 2: Evaluation of quasi-Newton methods using the Rosenbrock test case for staring points x_0, $10x_0$, and $100x_0$, with $x_0 = [x_{i=odd}, x_{i=even}] = [-1.2, 1]$, and for different dimensions N. The three best results per case are marked in bold.

	Dimension N	10^3						10^4						10^5					
	Starting point	x_0		$10x_0$		10^2x_0		x_0		$10x_0$		10^2x_0		x_0		$10x_0$		10^2x_0	
		it	$F+G$	*it*	$F+G$	*it*	$F+G$	*it*	$F+G$	*it*	$F+G$	*it*	$F+G$	*it*	$F+G$	*it*	$F+G$	*it*	$F+G$
damped L-BFGS	Dennis and Schnabel	24	**63**	60	157	117	320	24	**63**	60	160	123	327	24	**63**	60	159	110	306
	Powell	41	**99**	47	**121**	83	**207**	41	**99**	48	**123**	82	**204**	41	**99**	48	**123**	81	**200**
	Bisection	41	117	76	220	97	301	41	117	73	212	109	315	41	117	82	234	107	319
L-BFGS	Moré–Thuente	34	**96**	48	**134**	88	**248**	34	**96**	48	**134**	87	**246**	34	**96**	48	**134**	89	**252**
	Nocedal-quadratic	37	124	50	172	106	326	38	126	50	173	104	320	38	126	49	168	103	310
	Nocedal-bisection	35	110	45	**126**	95	**266**	35	110	46	**128**	93	**266**	35	110	45	**126**	94	**288**
dTN+L-BFGS	Moré–Thuente	36	158	52	263	97	542	36	158	52	261	100	547	36	158	52	264	104	557
	Nocedal-quadratic	27	155	54	319	112	950	28	160	51	297	112	950	27	155	54	313	118	1021
	Nocedal-bisection	27	194	51	417	103	848	27	194	51	416	122	966	27	194	51	418	104	874

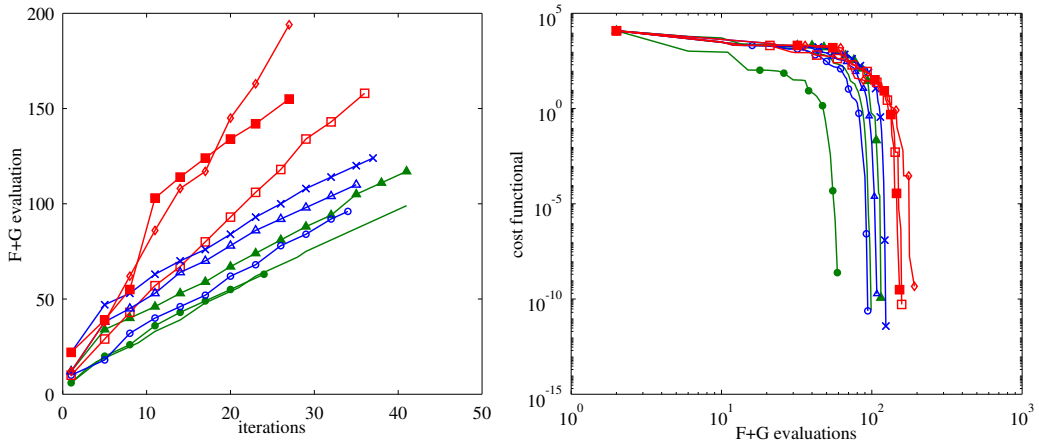

Figure 4: The cumulative sum of F+G evaluations during the Rosenbrock function optimization process (left) and the decrease in the objective functional with number of evaluations (right) in the damped L-BFGS with Dennis and Schnabel (•), Powell (—) and bisection (▲) line search methods; L-BFGS with Moré–Thuente (○), Nocedal bisection-quadratic (×) and Nocedal bisection (△) line search methods and discrete TN L-BFGS with Moré–Thuente (□), Nocedal bisection-quadratic (■) and Nocedal bisection (◊) algorithm. Initial point x_0 and dimension $N = 10^3$. Symbols are employed to differentiate lines.

iterations $\overline{it}$, and function plus gradient evaluations $\overline{F + G}$ are given in Table 3. First of all, we observe that in terms of convergence speed, these starting point are a lot more challenging than the points used in Table 2. Again the discrete Truncated Newton Method is outperformed by the other methods. Moreover, the methods that perform best on average, are damped L-BFGS with Dennis and Schnabel, damped L-BFGS with Powell, and the L-BFGS with Moré–Thuente line search. When comparing damped L-BFGS methods, and L-BFGS methods, we see that the latter sometimes require less iterations, but this does not always result in less function plus gradient evaluations, as the damped L-BFGS line search only requires one function evaluation per line-search iteration, while the L-BFGS method also requires a gradient evaluation for testing the second Wolfe condition.

Apart from number of iterations to convergence, also the cost-functional improvement when the algorithm is stopped before convergence is of relevance for DNS-based optimization. These optimization problems are often so expensive, that formal convergence is never

Table 3: The average number of iterations and sum of functional and gradient evaluations for the Rosenbrock function optimization using different quasi-Newton methods. Fifty random starting points are obtained from 50 different random sequences of size N lying in the interval $-2 \leq x_i \leq 2$. The random sequences are generated using the gnu Fortran90 standard random generator with seeds that correspond to $seed_j = int(10^7 random\_number(seed = 120368))$, and $j = 1 \cdots 50$. The three best results per case are marked in bold.

Algorithm	Line search	10^3		10^4		10^5	
		$\overline{it}$	$\overline{F+G}$	$\overline{it}$	$\overline{F+G}$	$\overline{it}$	$\overline{F+G}$
damped L-BFGS	Dennis and Schnabel	256.9	559.6	275.8	**598.4**	394.7	**845.4**
	Powell	249.7	**534.1**	288.9	**616.0**	530.7	**1124.9**
	Bisection	274.8	625.4	310.7	706.1	687.0	1519.0
L-BFGS	Moré–Thuente	237.2	**531.1**	272.4	**610.4**	417.9	**926.2**
	Nocedal-quadratic	248.0	**544.7**	278.4	618.4	604.4	1329.6
	Nocedal-bisection	253.1	559.8	292.9	648.1	568.2	1255.8
dTN+L-BFGS	Moré–Thuente	281.2	873.9	330.0	1084.0	569.3	2024.8
	Nocedal-quadratic	265.3	900.1	*316.9	*1225.2	617.2	2995.7
	Nocedal-bisection	287.9	1471.2	**318.4	**1637.6	609.4	3305.0

* The algorithm failed to converge for the 38th random starting point.

** The algorithm failed to converge for the 16th random starting point.

achieved before computational resources are exhausted. Therefore, in Table 4 the average cost function levels are compared for the different methods after 100, 150, and 200 function and gradient evaluations. To this end, starting points are based on the same random sequences as for Table 3. It is observed that again the damped L-BFGS with Dennis and Schnabel, damped L-BFGS with Powell, and the L-BFGS with Moré–Thuente algorithms perform best. Based on these results, and similar trends observed above, we select these three quasi-Newton algorithms for testing of DNS-based optimization in next subsection.

Finally, for cases in which unsteady adjoint-based optimization methods are applied to large scale problems the storage of the state variable at every time step and at every grid point during the forward integration of the primal equations may require excessive

Table 4: Cost function level after 100, 150, 200 functional plus gradient evaluations for the Rosenbrock test case starting from 50 different random sequences and $N = 10^4$. The same random sequences are used as for Table 3. The three best results per case are marked in bold.

Algorithm	Line search	The averaged cost functional ($\overline{J}$) after:		
		$F + G = 100$	$F + G = 150$	$F + G = 200$
damped L-BFGS	Dennis and Schnabel	940.3	**62.5**	**13.5**
	Powell	**699.0**	**64.59**	**17.2**
	Bisection	2927.1	362.9	67.7
L-BFGS	Moré–Thuente	**593.0**	69.2	**22.3**
	Nocedal-quadratic	995.3	98.8	32.2
	Nocedal-bisection	**604.1**	**68.3**	23.8
dTN+L-BFGS	Moré–Thuente	6677.1	3903.6	1813.1
	Nocedal-quadratic	7593.0	6887.2	6007.9
	Nocedal-bisection	6886.5	5170.9	3094.9

disk space. In these circumstances, a storage-saving technique based on check-pointing is often used [50–52]. In this case only a few intermediate flow solutions are stored, and the states are recomputed during the backward integration of the adjoint equations. For binomial checkpointing algorithms which are computationally optimal, the total amount of recomputing time amounts to one forward simulation [53]. Thus, in terms of computational cost, a gradient evaluation becomes twice as expensive as a standard function evaluation. When applying this situation to the cost analysis above (counting $F + 2G$ instead of $F + G$ for computational cost), we do not find any major changes (results not further shown here), and the best three algorithms remain the same. Since gradient evaluations become more expensive, the damped L-BFGS algorithm is favoured a bit more compared to counting $F + G$, but changes are small.

4.2. Optimal control of turbulent channel flow

Optimal control of turbulent channel flow is now investigated using three different quasi-Newton variants, i.e. the damped L-BFGS method with Powel line search, the damped L-BFGS method Dennis and Schnabel line search, and the L-BFGS method with Moré–Thuente line search. In addition, the Polak–Ribière method in combination with the Brent line-search is used as a reference, as this method is the current standard for DNS-based optimization. Two different optimal control cases are considered, respectively discussed in §4.2.1 and §4.2.2.

4.2.1. Optimal control of power extraction

First the optimal control case that focusses on increasing energy extraction by a distributed volume force is studied. Details of the problem formulation and computational set-up are provided in §2. For the time horizon of the optimal-control problem $T = T_F/2$ is selected, where $T_F = 0.8\delta/u_\tau$ is the average through-flow time of the domain. Given $f(x_3, t) \in \Omega_c \times [0, T]$, this leads to 6.4×10^4 degrees of freedom in the control space after discretization. The computational cost of one functional evaluation is approximately 40 minutes of wall time on 64 compute cores on an Ivy Bridge Xeon E5-2680v2 CPUs (2.8 GHz, 25 MB cache) processors; a similar time is required for an adjoint simulation. Finally, note that in practical DNS-based optimal control, the optimization problem over a time horizon T is followed by a control step Δt_c during which the optimal controls are applied, and then followed by a new optimization problem over $[\Delta t_c, \Delta t_c + T]$ etc. (cf., e.g., Ref. [1, 8] for details). Here we just perform optimization over one optimization interval only, as our focus is on optimization efficiency, and not so much on the optimal-control outcome when applied as a control algorithm.

First of all, in Figure 5, an overview is given of forward velocity fields $u(x, t)$ and adjoint fields $u^*(x, t)$ at time $t = 2T/3$ for different iterations during optimization using the damped L-BFGS method and Powell line search. In the first iteration (Figure 5a), the typical low-speed streaks characterizing turbulent channel flow close to the wall are observed. At later iterations (Figure 5c,e), the velocity in and below the forcing zone drops, and the turbulent

structures in this region change significantly.

The adjoint equations are linearized around the forward solution. The structure of these forward solutions is partly observed in the adjoint solutions (Figure 5b,d,f). As expected, the magnitude of the adjoint field in the first time steps is close to 0, which is determined by the initial condition ($u^*(x_1, x_2, x_3, 0) = 0$) derived from Eq. (15). As the backward flow evolves, the production of the adjoint energy in the interior of the fluid domain is increasing and the sensitivity of the objective functional with respect to the control is captured.

In Figure 6(a), the plane-averaged velocity field $\langle u_1 \rangle(x_3, t)$ (using $\langle \cdots \rangle$ to denote averaging over x_1 and x_2 directions) is shown as function of time, while Figure 6(a) shows $f(x_3, t)$ for the optimal controls obtained using a damped L-BFGS method with Powell line search, and converged up to 5×10^{-4}. It is observed that the mean velocity in the control region drops dramatically during the first part of the time interval, subsequently levels out at a level of $\langle u_1 \rangle / u_\tau \approx 6$, and finally drops again in the last stages of the time horizon. The related controls are relatively high in the initial stage, level out at a lower level in the intermediate stages, and increase dramatically near the end. This is typical for a finite-time optimal control problem. In particular the rise of control force near the end aims at extracting as much as possible the remaining kinetic energy from the system. At earlier time stages, this is not optimal, since the extracted power is proportional to the magnitude of the velocity.

We now proceed with a comparison of the three quasi-Newton methods in Figure 7. Results of the Polak–Ribière method are also added. The cumulative sum of the functional and gradient evaluations as function of the number of iterations (Fig. 7a) and the corresponding decrease rate of the objective functional (Fig. 7b) are shown. For this particular DNS-based optimal control problem, results indicate that the damped L-BFGS method is cheaper per iteration than the other methods that are tested. The combination of damped L-BFGS with Dennis and Schnabel line search requires only 86 functional and gradient evaluations, which is 36.3% less than the L-BFGS algorithm with Moré–Theunte line search, and 5.5% less than damped L-BFGS with Powell line search. All three methods require roughly the same number of iterations, but the damped L-BFGS methods require less simulations

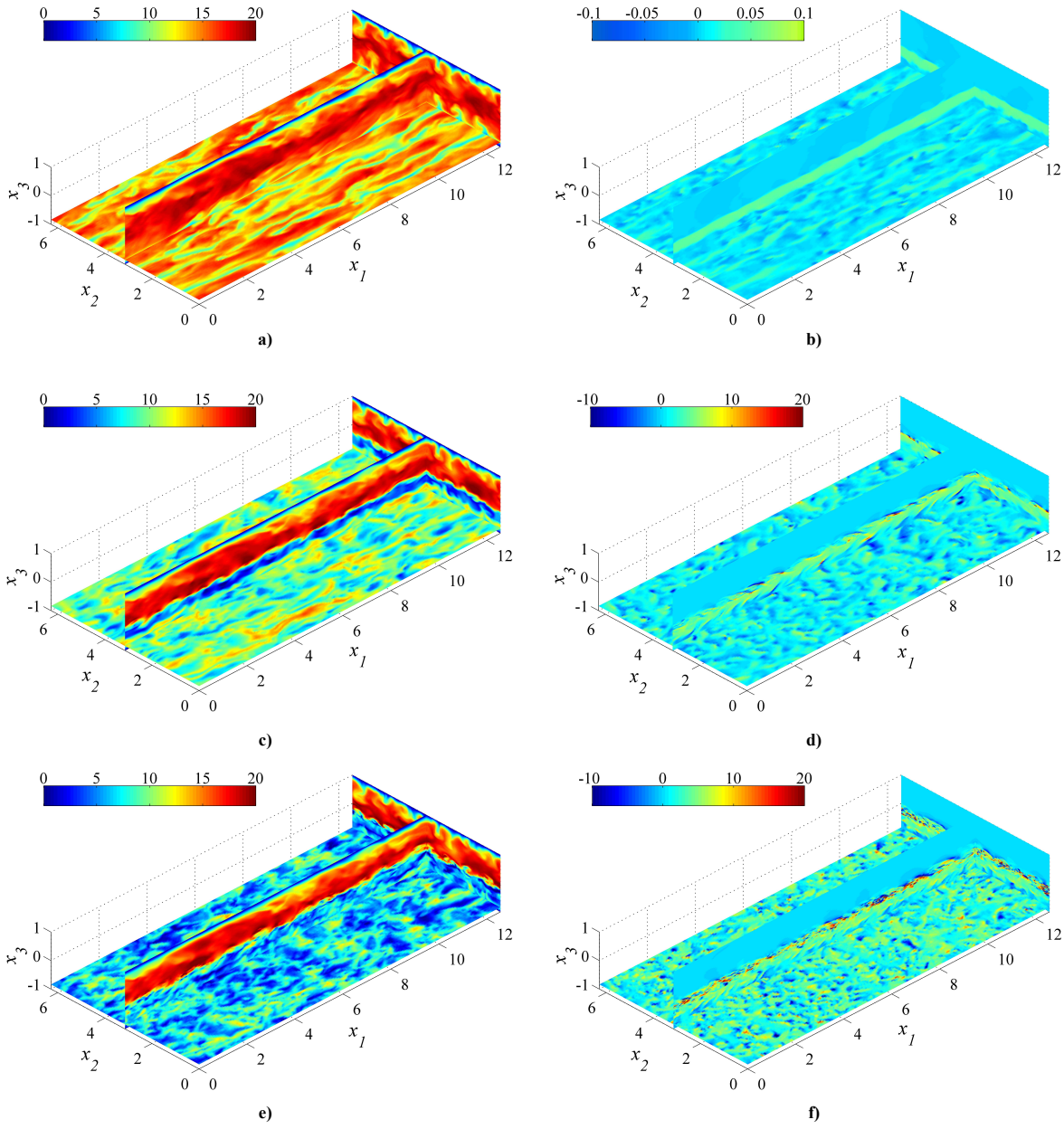

Figure 5: The contours of the instantaneous velocity field $u_1(x,t)$ (left) and adjoint field $u_1^*(x,t)$ (right) evaluated at time $t = 2T/3$ during different iterations of the optimization process: a)-b) $it_{opt} = 1$; c)-d) $it_{opt} = 7$; e)-f) $it_{opt} = 12$, using the damped L-BFGS method in combination with Powell line search. The control volume location is $L_{x_1} \times L_{x_2} \times [-0.8, -0.4]$

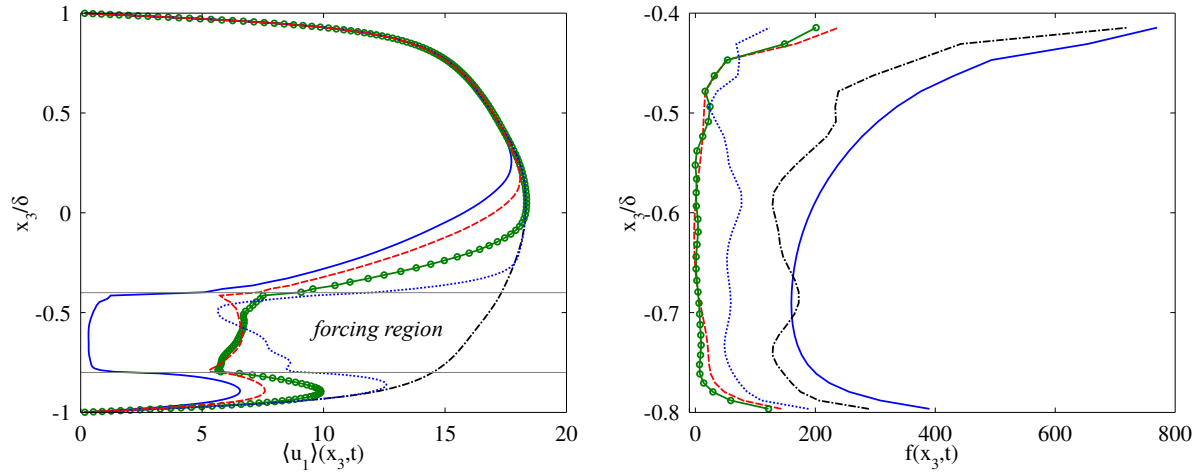

Figure 6: The plane-averaged velocity field $\langle u_1 \rangle(x_3, t)$ and the optimal controls $f(x_3, t)$ at time: $t = 0$ (-·-·-), $t = T/5$ (···), $t = T/2$ (○), $t = 4T/5$ (- - -) and $t = T$ (——). The control volume location is at $L_{x_1} \times L_{x_2} \times [-0.8\delta, -0.4\delta]$

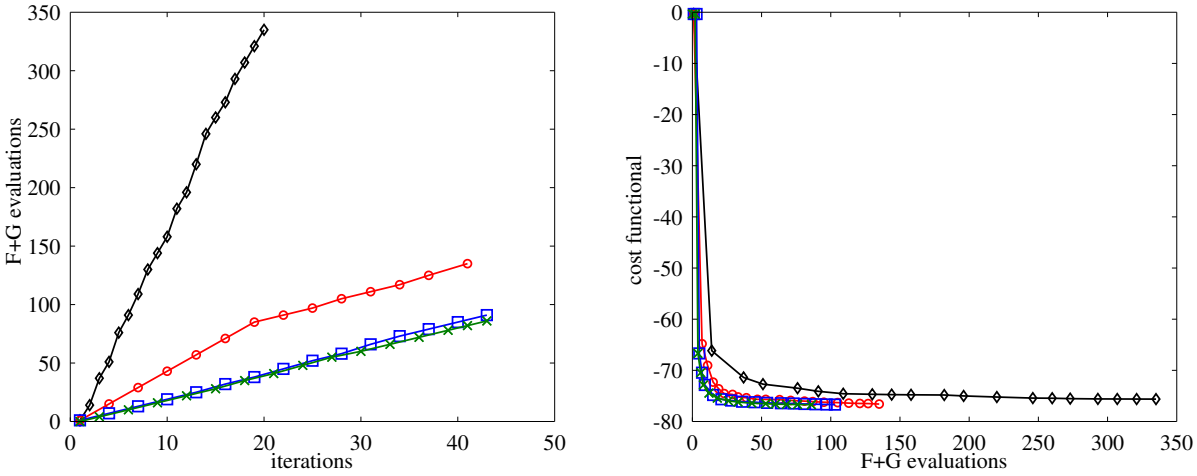

Figure 7: The cumulative sum of functional and gradient (F+G) evaluations per iteration (left) and the decrease of the objective functional $\hat{\mathcal{J}}$ with the F+G evaluations (right) for Polak–Ribière nonlinear conjugate gradient (◊), L-BFGS with Moré–Thuente line search (○), damped L-BFGS with Powell (□) and Dennis and Schnabel (×) line search methods.

per iteration (on average 2.04, and 2.12 evaluations for the Dennis and Schnabel, and the Powell line search, respectivelly) than L-BFGS with Moré–Thuente line-search (on average 3.29 evaluations). Finally, when comparing the quasi-Newton methods with Polak–Ribière, it is clear that the nonlinear conjugate gradient method requires a lot more functional and gradient evaluations, i.e. the algorithm was stopped after 340 evaluations before convergence was formally reached. Furthermore, if we consider stopping optimization early, e.g., after 40 functional and gradient estimations, we find that the gain in cost functional for the Polak–Ribière method is 6.5% less than for the (damped) L-BFGS methods.

In a variant of the optimal-control of energy extraction, we add a penalty term to the cost functional that corresponds to (cf. Eq. (1)):

$$\mu\mathcal{T}(f,u) = \mu\frac{\int\limits_{\Omega}[e(x,T)-e(x,0)]^2\mathrm{d}x}{\int\limits_{\Omega}e(x,0)\mathrm{d}x} \tag{37}$$

and with $e(x,t) = u(x,t)\cdot u(x,t)/2$. The additional term penalizes a decrease of overall kinetic energy in the system and is relevant in the context of wind-farm optimal control (cf. Ref. [8]). The penalization factor μ is dimensionless, and we use $\mu = 0.8$.

Results are shown in Figure 8. Compared to the unpenalized case above, the optimization converges much faster, i.e. within 20 to 40 evaluations. For this case, it is observed that the Polak–Ribière method remains roughly in the same league as the quasi-Newton methods in terms of cost. When looking at the quasi-Newton methods, it is found that for $\hat{\mathcal{J}}$ damped L-BFGS requires 24-25 $F+G$ evaluations (for both line search methods), while L-BFGS requires 34 $F+G$. In case we consider stopping optimization before convergence, i.e. after 5 to 10 evaluations, it is observed that the damped L-BFGS slightly outperforms the L-BFGS methods, and also gains over the conjugate-gradient method are significant. Note that in Figure 8(right), the damped L-BFGS method performs very poorly in the first iteration (though this is compensated for afterwards). This is related to the initialization of the algorithm. For this, $H_k^0 = \gamma_k I$ with $\gamma_{k=1} = 1$ is used (cf. §3.2). The convergence properties of the (damped) L-BFGS algorithm are insensitive to this initial matrix [54]. However, $H_k^0 = I$ can lead to poor scaling in the first iteration, so that the search direction is not a

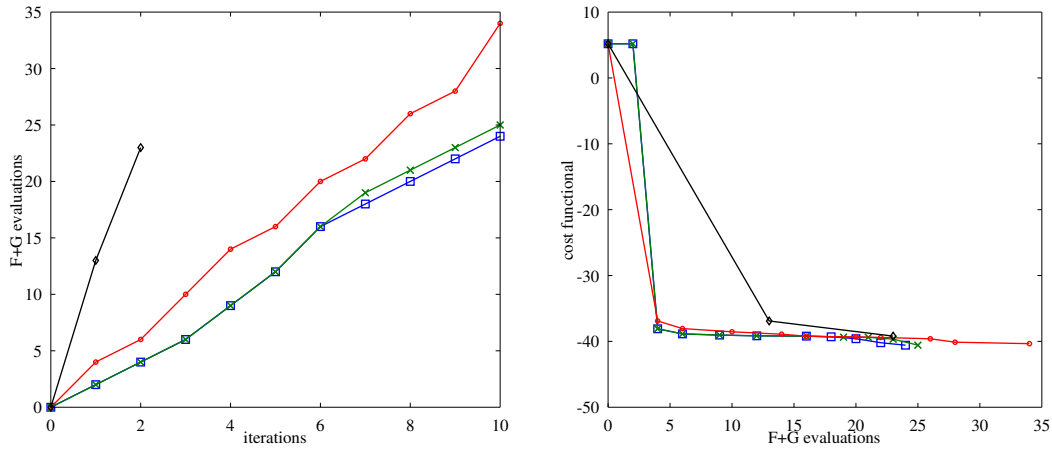

Figure 8: The cumulative sum of functional and gradient (F+G) evaluations per iteration (left) and the decrease of the objective functional(right) in L-BFGS with Moré–Thuente line search method (○), damped L-BFGS with Powell (□) and Dennis and Schnabel (×) line search methods and Polak–Ribière nonlinear conjugate gradient (◊) for penalized cost function with $\mu = 0.8$.

strong decrease direction. Moreover, in the damped L-BFGS methods, only the sufficient decrease condition is verified, so that the variation of the cost functional in these first steps can remain very small (requiring the second Wolfe condition as in the L-BFGS algorithm can lead to larger step sizes). As is appreciated from the figure, the curvature information in the Hessian improves very fast, so that this problem does not persist after the first iteration.

4.2.2. Optimal control of turbulent kinetic energy

A second optimal-control case aims at the minimization of turbulent kinetic energy. The simulation details and the control configuration are discussed in §2. The optimal control time horizon is again selected at $T = T_f/2$, leading to 1.6×10^5 degrees of freedom for the control force $f(x_1, t)$. Both the forward simulations and the adjoint simulations require approximately 60 minutes of wall time on 32 computes cores using the same processor architecture as discussed above. Again, we only consider the optimization problem of one leg of a typical receding-horizon optimal control (cf. discussion above), since we are mainly interested in the convergence history of the optimization here.

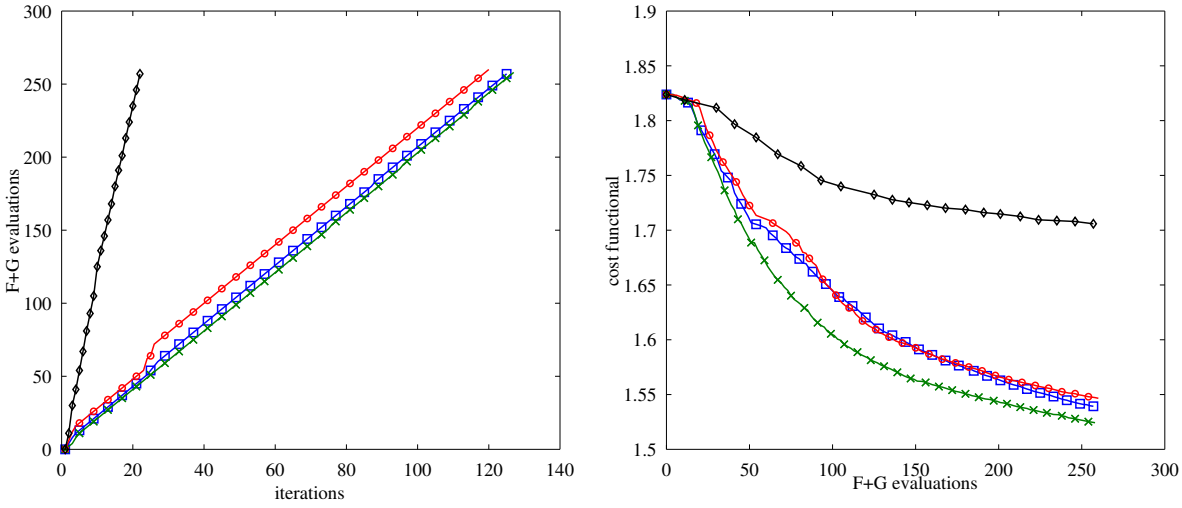

Figure 9: The cumulative sum of functional and gradient (F+G) evaluations per iteration (left) and the decrease of the objective functional with the F+G evaluations (right) for Polak–Ribière nonlinear conjugate gradient (◊), L-BFGS with Moré–Thuente line search (○), damped L-BFGS with Powell (□) and Dennis and Schnabel (×) line search methods. Symbols are employed to differentiate lines.

Figure 9 shows the cumulative number of evaluations per iteration for the different optimization methods, as well as the decrease of the cost functional versus the sum of functional and gradient estimations. Compared with the previous optimal control case, the current test case appears to be much more challenging, requiring many more iterations. In fact, in view of computational costs, we stopped the iterations after 260 evaluations. In terms of improvement of the cost functional, the L-BFGS and damped L-BFGS with Powell line search lead to decreases of 18%, and 18.5% respectively. The damped L-BFGS version with the Dennis and Schnabel line search leads to a reduction of 19.7%, whereas the nonlinear conjugate gradient is clearly outperformed by the other methods, achieving only 7% in the same amount of DNS or adjoint DNS evaluations. Finally, in terms of drag reduction, we observe a reduce in skin friction of 10.3%. Note however, that we have considered only one time horizon, and a sequence of time horizons should be considered in a moving-horizon framework for the evaluation of the overall drag-reduction potential (cf., e.g., the approach in Ref. [1]).

5. Conclusions

In the current work we presented different quasi-Newton methods for unconstrained optimization in problems that use direct numerical simulations of turbulent flows. The performance of these methods was evaluated in terms of functional and gradient requirements for convergence, as well as relative cost functional improvement. These are of crucial importance for DNS-based problems, as both simulations and gradient evaluations are extremely expensive. Three quasi-Newton methods were investigated, i.e. the limited-memory BFGS methods, the damped L-BFGS method, and the discrete truncated Newton method. As a point of reference, the current standard in DNS-based optimization, namely the Polak–Ribière non-linear conjugate-gradient method was also investigated.

The quasi-Newton methods were combined with three different line-search methods, based on bisection, quadratic interpolation, and cubic interpolation. Firstly, the extend Rosenbrock function was used to select the three most performing combinations. Based on this analysis, damped L-BFGS with quadratic and cubic line search, and L-BFGS with cubic line search were selected for further testing in DNS-based optimal control of turbulent channel flow. To that end, two different type of optimal control problems were considered, one related to increase of energy extraction by a distributed volume force in the channel, the other related to drag reduction by wall forcing.

Overall, we found the damped L-BFGS with cubic line search the best method. Damped L-BFGS methods are in principle designed for constrained optimization problems [5], and yield a less accurate Hessian estimation of the cost functional than L-BFGS methods. We observed that this leads to slightly more iterations for convergence. However, the line search in the damped L-BFGS method only needs to satisfy the Armijo condition, while in L-BFGS methods the Wolfe conditions also need to be met. This leads to a larger overhead per iteration for the L-BFGS method. Differences are not always large, but sometimes significant, e.g., for the first DNS-based optimization case damped L-BFGS required 36% less DNS and adjoint DNS simulations than L-BFGS. The difference when applying the Polak–Ribière conjugate gradient is much larger, i.e., more than a factor 4 in some cases.

Also, when looking at the relative level of cost functional improvement when iterations are stopped early, we found that the damped L-BFGS method with cubic line search performed best. This can be quite relevant, since in DNS-based optimization, algorithms are not always formally converged, as computational resources can be excessive.

Finally, in the current study, we applied different quasi-Newton methods to two DNS-based optimization cases that are relevant for a range of optimal-control problems in wall-bounded flows. However, in turbulence, many more optimization problems exist, related to different types of flows (e.g. jets, mixing layers, ...), different types of objectives (noise reduction, mixing efficiency, ...), and different types of control variables (geometrical shape and topology, various types of actuators, ...). It will be interesting in further research to test the quasi-Newton methods discussed here for DNS-based optimization in these applications.

6. Acknowledgements

The authors acknowledge support from OPTEC (OPTimization in Engineering Center of Excellence, KU Leuven), which is funded by the KU Leuven Research Council under grant no PFV/10/002. Simulations were performed on the computing infrastructure of the VSC Flemish Supercomputer Center, funded by the Hercules Foundation and the Flemish Government.

7. Appendix A

The discrete truncated Newton algorithm in combination with the L-BFGS method is given by Algorithm 1 and 2.

8. Appendix B

The second phase in the bisection-quadratic line search method following the Nocedal procedure ([18], from page 61) is given in Algorithm 3.

method=L-BFGS;

while *convergence=.FALSE.* **do**

1. find the search direction ($\nabla^2 f(x_k)p_k = -\nabla f(x_k)$) Call Inner Loop (cf. Algorithm 2);
2. compute the step length α_k;
3. update $x_{k+1} = x_k + \alpha_k p_k$;
4. calculate and store the pairs: $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$;
5. check the convergence $||\nabla f(x_{k+1})|| \leq 10^{-4}$;
6. **if** *the CG iterations generate direction of negative curvature or the step length is not the Newton step* $\alpha_k \neq 1$ **then**

   method=L-BFGS;

   else

   method=dTN;

   end

end

Algorithm 1: Outer loop

$r^{(0)} = d^{(0)} = -\nabla f(x_k)$, itcgmax=2 (the maximum number of CG iterations);

$\epsilon = \min(0.5, \sqrt{||\nabla f(x_k)||})||\nabla f(x_k)||$ ([18] pp. 169);

for *j=1,itcgmax* **do**

  if $j > 1$ **then**

    decide if $v^{(j-1)}$ and $A_k v^{(j-1)}$ is to be saved and update H_k

  end

  if *k=1* **then**

    $z^{(j-1)} = r^{(j-1)}$;

  else

    $z^{(j-1)} = H_k r^{(j-1)}$ compute with two-loop recursion algorithm [18];

  end

  if *method=L-BFGS* **then**

    return with $p_k = z^{(0)}$

  end

  if $||r^{(j-1)}|| < \epsilon$ **then return** with $p_k = d^{(j-1)}$;

  $\rho_{j-1} = (r^{(j-1)})^T z^{(j-1)}$;

  if *j=1* **then**

    $v^{(j)} = -z^{(j-1)}$;

  else

    $\beta_{j-1} = \frac{\rho_{j-1}}{\rho_{j-2}}$;

    $v^{(j)} = -z^{(j-1)} + \beta_{j-1} v^{(j-1)}$;

  end

  approximate the product $A_k v^{(j)}$ with finite difference;

  if $(v^{(j)})^T A_k v^{(j)} < \delta$ **then return** with $p_k = d^{(j-1)}$;

  $\alpha_j = \frac{\rho_{j-1}}{(v^{(j)})^T A_k v^{(j)}}$;

  $d^{(j)} = d^{(j-1)} + \alpha_j v^{(j)}$;

  $r^{(j)} = r^{(j-1)} + \alpha_j A_k v^{(j)}$;

  if *j=itcgmax* **then return** with $p_k = d^{(itcgmax)}$;

end

Algorithm 2: The inner loop in algorithm 1

check if $\alpha_{lo} < \alpha_{hi}$, it=0;

repeat

$\delta\alpha = \alpha_{hi} - \alpha_{lo}$;

$b = f(\alpha_{hi}) - f(\alpha_{lo}) - \delta\alpha slope(\alpha_{lo})$;

if $b \leq 0$ *.or.* $it \geq 5$ **then**

$\alpha = \alpha_{lo} + \frac{\delta\alpha}{2}$;

else

$\alpha = \alpha_{lo} - \frac{slope(\alpha_{lo})\delta\alpha^2}{2b}$;

end

evaluate $f(\alpha)$;

if $f(\alpha) > f(0) + 10^{-4}\alpha slope(0)$*.or.*$f(\alpha) \geq f(\alpha_{lo})$ **then**

$\alpha_{hi} = \alpha$; $f(\alpha_{hi}) = f(\alpha)$

else

evaluate $\nabla f(\alpha)$ and $slope(\alpha) = (\nabla f(\alpha))^T p_k$;

if $slope(\alpha) \leq -0.9 slope(0)$ **then**

return with $\alpha^* = \alpha$

end

if $slope(\alpha) \geq 0$ **then**

$\alpha_{hi} = \alpha_{lo}$; $f(\alpha_{hi}) = f(\alpha_{lo})$

end

$\alpha_{lo} = \alpha$;

$f(\alpha_{lo}) = f(\alpha)$;

$slope(\alpha_{lo}) = slope(\alpha)$

end

it=it+1;

until *find* α^*;

Algorithm 3: Second phase in the Nocedal line search algorithm ([18], from page 61) to find a step length α^* in interval $[\alpha_{lo}, \alpha_{hi}]$

[1] T. R. Bewley, P. Moin, R. Temam, DNS-based predictive control of turbulence: an optimal benchmark for feedback algorithms, Journal of Fluid Mechanics 447 (2001) 179–225.

[2] J. Kim, D. J. Bodony, J. B. Freund, Adjoint-based control of loud events in a turbulent jet, Journal of Fluid Mechanics 741 (2014) 28–59.

[3] J. B. Freund, Adjoint-based optimization for understanding and suppressing jet noise, Journal of Sound and Vibration 330 (17) (2011) 4114–4122.

[4] M. Wei, J. B. Freund, A noise-controlled free shear flow, Journal of Fluid Mechanics 546 (2006) 123–152.

[5] H. Badreddine, S. Vandewalle, J. Meyers, Sequential quadratic programming (SQP) for optimal control in direct numerical simulation of turbulent flow, Journal of Computational Physics 256 (2014) 1–16.

[6] S. Delport, M. Baelmans, J. Meyers, Constrained optimization of turbulent mixing-layer evolution, Journal of Turbulence 10 (18) (2009) 1–26.

[7] S. Delport, M. Baelmans, J. Meyers, Maximizing dissipation in a turbulent shear flow by optimal control of its initial state, Physics of Fluids 23 (4) (2011) 045105.

[8] J. P. Goit, J. Meyers, Optimal control of energy extraction in wind-farm boundary layers, Journal of Fluid Mechanics 768 (2015) 5–50.

[9] D. C. Liu, J. Nocedal, On the limited memory BFGS method for large scale optimization, Mathematical programming 45 (1-3) (1989) 503–528.

[10] S. G. Nash, J. Nocedal, A numerical study of the limited memory BFGS method and the truncated-newton method for large scale optimization, SIAM Journal on Optimization 1 (3) (1991) 358–372.

[11] F. Abergel, R. Temam, On some control problems in fluid mechanics, Theoretical and Computational Fluid Dynamics 1 (6) (1990) 303–325.

[12] M. Gunzburger, Adjoint equation-based methods for control problems in incompressible, viscous flows, Flow, Turbulence and Combustion 65 (3-4) (2000) 249–272.

[13] A. Jameson, L. Martinelli, N. Pierce, Optimum aerodynamic design using the navier–stokes equations, Theoretical and computational fluid dynamics 10 (1-4) (1998) 213–237.

[14] D. Thevenin, G. Janiga, Optimization and Computational Fluid Dynamics, Springer Berlin Heidelberg, 2008.

[15] M. Hinze, K. Kunisch, Second order methods for optimal control of time-dependent fluid flow, SIAM Journal on Control and Optimization 40 (3) (2001) 925–946.

[16] A. Borzi, V. Schulz, Computational optimization of systems governed by partial differential equations, SIAM, Philadelphia, 2012.

[17] F. Tröltzsch, Optimal control of partial differential equations: theory, methods, and applications, Grad. Stud. Math. 112, American Mathematical Society, Providence, RI, 2010.

[18] J. Nocedal, S. J. Wright, Numerical optimization, Springer New York, 2006.

[19] W. W. Hager, H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, SIAM Journal on Optimization 16 (1) (2005) 170–192.

[20] Y.-H. Dai, Y. Yuan, A nonlinear conjugate gradient method with a strong global convergence property, SIAM Journal on Optimization 10 (1) (1999) 177–182.

[21] S. G. Nash, A survey of truncated-newton methods, Journal of Computational and Applied Mathematics 124 (1) (2000) 45–59.

[22] S. G. Nash, Preconditioning of truncated-newton methods, SIAM Journal on Scientific and Statistical Computing 6 (3) (1985) 599–616.

[23] M. J. Powell, A fast algorithm for nonlinearly constrained optimization calculations, in: Numerical analysis, Springer Berlin Heidelberg, 1978, pp. 144–157.

[24] J. J. Moré, D. J. Thuente, Line search algorithms with guaranteed sufficient decrease, ACM Transactions on Mathematical Software (TOMS) 20 (3) (1994) 286–307.

[25] J. E. Dennis Jr, R. B. Schnabel, Numerical methods for unconstrained optimization and nonlinear equations, Vol. 16, SIAM, 1996.

[26] X. Zou, I. M. Navon, M. Berger, K. H. Phua, T. Schlick, F.-X. Le Dimet, Numerical experience with limited-memory quasi-newton and truncated newton methods, SIAM Journal on Optimization 3 (3) (1993) 582–608.

[27] H. Choi, P. Moin, J. Kim, Active turbulence control for drag reduction in wall-bounded flows, Journal of Fluid Mechanics 262 (1994) 75–110.

[28] H. Mamori, K. Fukagata, Drag reduction effect by a wave-like wall-normal body force in a turbulent channel flow, Physics of Fluids (1994-present) 26 (11) (2014) 115104.

[29] M. Calaf, C. Meneveau, J. Meyers, Large eddy simulation study of fully developed wind-turbine array boundary layers, Physics of Fluids (1994-present) 22 (1) (2010) 015110.

[30] S. S. Collis, Y. Chang, On the use of les with a dynamic subgrid-scale model for optimal control of wall bounded turbulence, in: Recent Advances in DNS and LES, Springer Netherlands, 1999, pp. 99–110.

[31] H. Choi, M. Hinze, K. Kunisch, Instantaneous control of backward-facing step flows, Applied Numerical Mathematics 31 (2) (1999) 133–158.

[32] M. D. Gunzburger, Perspectives in flow control and optimization, Vol. 5, SIAM, Philadelphia, 2003.

[33] M. B. Giles, N. A. Pierce, An introduction to the adjoint approach to design, Flow, turbulence and combustion 65 (3-4) (2000) 393–415.

[34] R. Vishnampet, D. J. Bodony, J. B. Freund, A practical discrete-adjoint method for high-fidelity compressible turbulence simulations, Journal of Computational Physics 285 (2015) 173–192.

[35] A. Carnarius, F. Thiele, E. Oezkaya, N. R. Gauger, Adjoint approaches for optimal flow control, In: 5th Flow Control Conference, Chicago, Illinois. AIAA-2010-5088.

[36] S. Nadarajah, A. Jameson, A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization, AIAA-2000-0667.

[37] J. E. Hicken, D. W. Zingg, Dual consistency and functional accuracy: a finite-difference perspective, Journal of Computational Physics 256 (2014) 161–182.

[38] R. Verstappen, A. Veldman, Symmetry-preserving discretization of turbulent flow, Journal of Computational Physics 187 (1) (2003) 343–368.

[39] C. Canuto, M. Y. Hussaini, A. Quarteroni, T. A. Zang, Spectral Methods, Fundamentals in Single Domains, Springer Berlin Heidelberg, 2006.

[40] J. Kim, P. Moin, R. Moser, Turbulence statistics in fully developed channel flow at low reynolds number, Journal of Fluid Mechanics 177 (1987) 133–166.

[41] R. D. Moser, J. Kim, N. N. Mansour, Direct numerical simulation of turbulent channel flow up to Re= 590, Phys. Fluids 11 (4) (1999) 943–945.

[42] M. Quadrio, Drag reduction in turbulent boundary layers by in-plane wall motion, Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 369 (1940) (2011) 1428–1442.

[43] D. G. Luenberger, Y. Ye, Linear and nonlinear programming, Vol. 116, Springer Science & Business Media, 2008.

[44] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, Numerical recipes in Fortran 77 and Fortran 90, Cambridge University Press Cambridge, 1996.

[45] C. T. Kelley, Iterative methods for optimization, Vol. 18, SIAM, 1999.

[46] R. H. Byrd, J. Nocedal, C. Zhu, Towards a discrete newton method with memory for large-scale optimization, Nonlinear Optimization and Applications (1996) 1–12.

[47] J. L. Morales, J. Nocedal, Enriched methods for large-scale unconstrained optimization, Computational Optimization and Applications 21 (2) (2002) 143–154.

[48] S. Kok, C. Sandrock, Locating and characterizing the stationary points of the extended rosenbrock function, Evolutionary computation 17 (3) (2009) 437–453.

[49] J. J. Moré, B. S. Garbow, K. E. Hillstrom, Testing unconstrained optimization software, ACM Transactions on Mathematical Software (TOMS) 7 (1) (1981) 17–41.

[50] A. Nemili, E. Özkaya, N. R. Gauger, A. Carnarius, F. Thiele, A discrete adjoint approach for unsteady optimal flow control, in: New Results in Numerical and Experimental Fluid Mechanics VIII, Springer, 2013, pp. 447–455.

[51] Q. Wang, P. Moin, G. Iaccarino, Minimal repetition dynamic checkpointing algorithm for unsteady adjoint calculation, SIAM Journal on Scientific Computing 31 (4) (2009) 2549–2567.

[52] R. Roth, S. Ulbrich, A discrete adjoint approach for the optimization of unsteady turbulent flows, Flow, turbulence and combustion 90 (4) (2013) 763–783.

[53] M. Hinze, A. Walther, J. Sternberg, An optimal memory-reduced procedure for calculating adjoints of the instationary navier-stokes equations, Optimal Control Applications and Methods 27 (1) (2006) 19–40.

[54] A. Griewank, The local convergence of broyden-like methods on lipschitzian problems in hilbert spaces, SIAM Journal on Numerical Analysis 24 (3) (1987) 684–705.